**复旦大学计算机科学技术学院**

**2024-2025学年第一学期期末论文课程评分表**

**课程名称： 计算机可视化 课程代码：COMP130174.01**

**开课院系： 计算机科学技术学院**

**学生姓名：洪成勋 学号：22300240021 专业：计算机科学与技术**

**论文名称：Github关注量Top200用户数据的可视化分析及实现**

|  |
| --- |
| **论文评语（教师填写）：**  **任课教师签名：**  **日 期：** |

**成绩：**

**Github关注量Top200用户数据的可视化分析及实现**

洪成勋 22300240021

【摘要】

随着开源文化在全球范围内蓬勃发展，Github 作为最大的开源代码托管平台之一，其用户数据蕴含着丰富的信息。本设计聚焦于 Github 关注量 Top200 用户，旨在深入挖掘与剖析这些高关注度用户群体的特征及行为模式。通过自主编写的网络爬虫工具精准地从 Github 官网抓取关键个人信息，涵盖用户名、地理位置、粉丝数、贡献数等多元维度，并基于数据清洗与整理流程，运用 Python 数据处理库、AI代码助手Cursor及专业可视化工具，精心绘制一系列丰富且极具洞察性的图表，如全球视野下的用户分布地图、多变量对比的柱状与折线图、关联关系映射的桑基图和旭日图等。设计成果不仅直观呈现高关注用户的静态画像，更动态揭示其在年份、语言使用及组织参与等维度的演变趋势，可为开源社区生态研究、开发者行为分析及平台优化策略制定提供全新视角与量化依据。

（本文相关代码仓库：https://github.com/hcxxxxxx/Visualization）

【正文】

1. 相关技术与理论基础
2. 数据获取机制

该部分旨在精准获取 Github 关注量前 200 用户的详细个人信息，为后续深入的可视化分析奠定坚实的数据基础。数据获取过程综合运用了多种技术手段，包括 requests 库、BeautifulSoup 库以及 Selenium 库，巧妙协同以克服 Github 页面动态加载及数据分散性难题，确保信息抓取的完整性与准确性。

数据抓取过程集成在fetch.py文件中，抓取步骤大致分为：目标URL构建与请求，页面解析与用户信息初筛，动态页面加载与关键数据抓取，以及补充信息采集与数据整合。抓取的数据放入文件person\_data\_top200.json中。

1. 可视化技术选型与原理

本设计对比了主流可视化库（Matplotlib、Seaborn、Plotly 等）的优劣势，最终依据不同图表类型（地理、关系、分布等）选定了Streamlit作为主框架，并结合了Pandas、Plotly绘制可视化图像。

可视化类型选择方面，本设计针对各类数据信息的关系和特性，选择了不同的图表结构和可视化形式。例如，采用柱形图、折线图阐述粉丝数与贡献量关联模式、不同年份用户流入流出趋势等；采用桑基图展示组织-语言资源流动原理、旭日图嵌套结构呈现国家-语言层次关系、世界地图展示热点用户分布；采用3D坐标图布局用户多属性向量，对用户数据进行多维观察和分析。

1. AI代码助手

本设计中全程借助AI代码助手Cursor进行智能代码推荐、AI问答以及代码调试和修改。部分关键对话与代码段落将在后文中呈现。

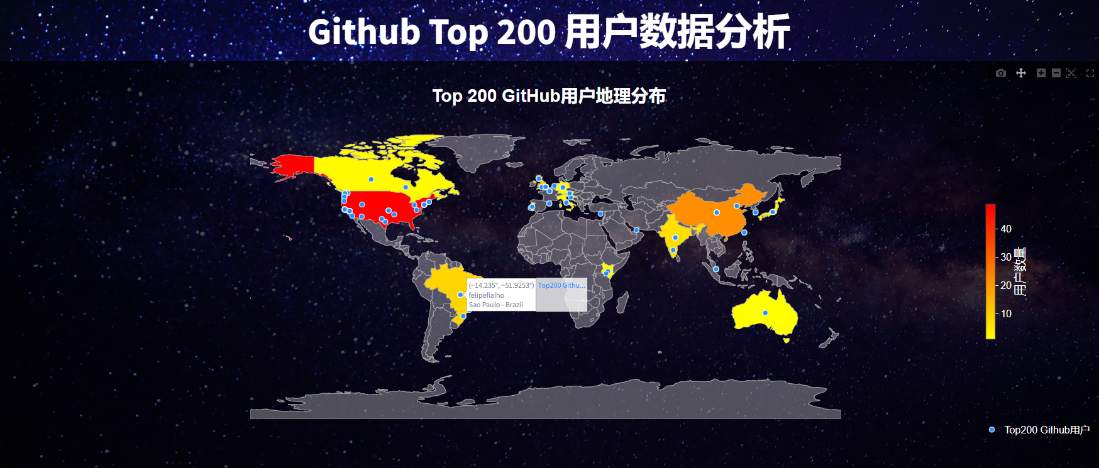
1. 数据收集与预处理
2. 数据来源

本研究的数据来源于对Github平台的爬取，获取了关注量Top200用户的多方面信息，包括但不限于用户名称、关注者数量、贡献数、仓库数量、编程语言、加入年份、地理位置等。

1. 数据清洗
2. 缺失值处理：对数据集中存在的缺失值进行识别和处理。对于部分关键信息缺失严重的记录，考虑予以删除；对于部分可补充或不影响整体分析的缺失值，采用合理的填充策略，如使用平均值、中位数或特定的默认值进行填充。
3. 异常值处理：通过数据分布分析和业务规则判断，识别并处理异常值。例如，对于明显偏离正常范围的关注者数量或贡献数，检查数据的准确性，如确为异常则进行修正或删除。
4. 数据格式转换：将日期格式的数据转换为统一的格式，以便后续按时间维度进行分析；对数值型数据进行类型转换，确保数据的一致性和准确性，如将字符串类型的数字转换为合适的数值类型。
5. 数据转换
6. 地理信息处理：将用户的地理位置信息转换为经纬度坐标，以便在地图可视化中准确展示用户分布。同时，对国家和地区信息进行标准化处理，建立统一的映射关系，便于统计和分析用户的国家分布情况。
7. 数值转换：对关注者数量、贡献数等数值型数据进行单位统一和量级转换，如将以“k”“m”为单位的数值转换为具体数字，以便进行准确的计算和比较。
8. 可视化方法与实现
9. 可视化库的选择

本项目选用了Plotly和Streamlit等Python可视化库。Plotly提供了丰富的交互式图表类型，如地图、柱状图、饼图、散点图、桑基图、旭日图等，能够满足多维度数据可视化的需求，并且生成的图表具有良好的交互性和美观性。Streamlit则用于构建Web应用，方便将可视化结果以直观的界面形式展示出来，同时提供了简洁的开发方式，便于快速迭代和部署。

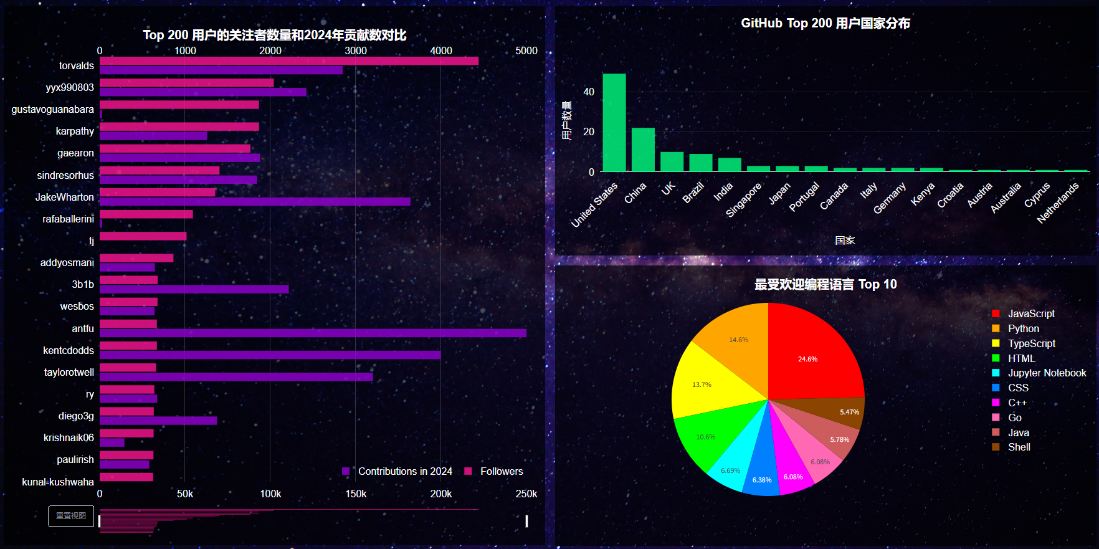
1. 具体可视化图表及实现
2. 世界地图（chart\_01）

·功能：展示Top200 Github用户在全球的地理分布，通过热力图直观呈现不同国家和地区的用户数量分布情况，同时在地图上标记用户的具体位置。·实现方式：使用Plotly的Choropleth地图类型，根据用户所在国家的统计数据绘制热力图，颜色深浅表示用户数量的多少。通过自定义颜色比例尺，从黄色到红色渐变，清晰展示用户数量的相对差异。对于用户位置标记，使用Scattergeo类型，以蓝色标记点显示用户所在经纬度位置，并在悬停时显示用户姓名和具体位置信息。

(chart1：世界地图)

1. 关注者数量和贡献数对比（chart\_02）

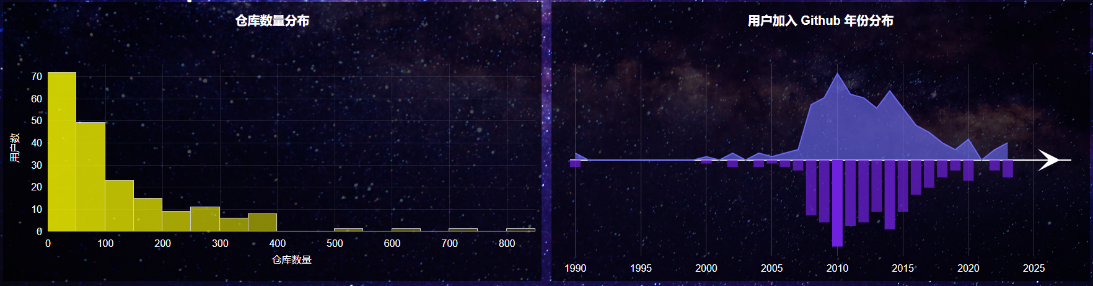
·功能：以水平双向柱状图对比Top200用户的关注者数量和2024年贡献数，帮助分析用户在社交影响力和近期贡献活跃度方面的关系。·实现方式：使用Plotly的Bar图表类型，分别绘制关注者数量和贡献数的柱状图，通过设置不同的颜色和x轴范围，清晰区分两者。为了增强交互性，添加了底部滑块，方便用户缩放查看数据范围；同时设置了重置按钮，使用户能够快速恢复初始视图。

1. 用户国家分布（chart\_03）·功能：以柱状图展示Github Top200用户的国家分布情况，直观呈现不同国家的用户数量占比。·实现方式：统计每个国家的用户数量，使用Plotly的Bar图表绘制柱状图，x轴为国家名称，y轴为用户数量。通过设置x轴标签倾斜角度，避免标签重叠；添加网格线，增强图表可读性。
2. 最受欢迎语言Top10（chart\_04）·功能：以饼图展示Top200用户中最受欢迎的前10种编程语言及其占比，反映用户的技术偏好。·实现方式：对用户使用的编程语言进行统计，选取使用频率最高的前10种语言，使用Plotly的Pie图表绘制饼图，每种语言使用不同的鲜艳颜色区分，使图表更加直观。

（chart2：关注者数量和贡献数对比

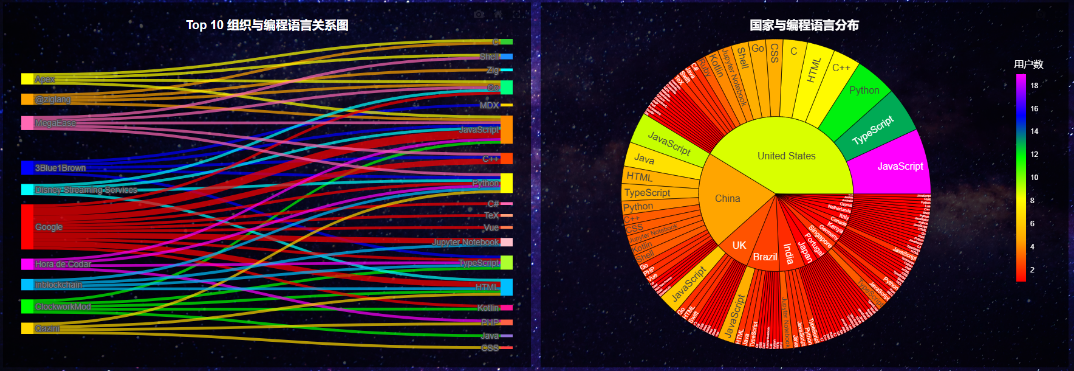
chart3：用户国家分布

chart4：最受欢迎语言Top10）

1. 仓库数量分布（chart\_05）·功能：以直方图展示Top200用户的仓库数量分布情况，帮助了解用户在项目创建方面的活跃度差异。·实现方式：使用Plotly的Histogram图表类型，将仓库数量数据划分为多个区间（bins），统计每个区间内的用户数量，以不同透明度的黄色标记直方图的柱子，展示仓库数量的分布趋势。
2. 加入Github年份分布（chart\_06）·功能：通过折线图和矩形图组合的方式展示用户加入Github的年份分布，包括每年加入的用户数量以及总体趋势，同时通过颜色深浅表示用户数量的相对多少。·实现方式：统计每年加入Github的用户数量，使用Plotly的Scatter和Scatter（filled）图表类型，绘制折线图表示趋势，以不同颜色和透明度的矩形填充表示每年的用户数量，颜色越深表示用户数量越多。添加箭头指示趋势方向，增强图表的可视化效果。

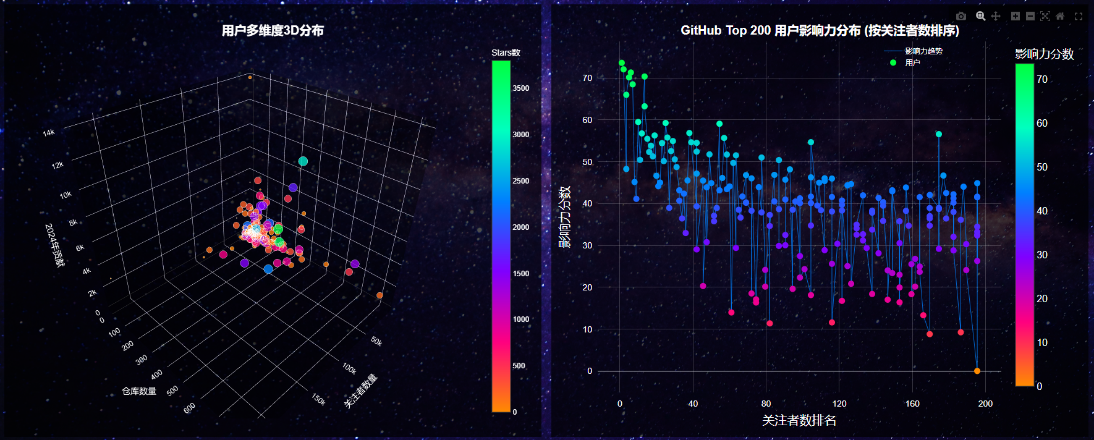
（chart5：仓库数量分布

chart6：加入Github年份分布）

1. Top10组织与编程语言关系（chart\_07）·功能：以桑基图展示Top10组织与编程语言之间的关系，包括组织使用的编程语言种类以及编程语言在不同组织中的应用情况，同时通过颜色表示节点的连接数多少。·实现方式：统计Top10组织使用的编程语言及相关数量，构建桑基图的节点和链接数据。根据节点的连接数计算颜色深浅，使用Plotly的Sankey图表类型绘制桑基图，组织节点和语言节点使用不同的颜色系列区分，链接使用半透明颜色显示，增强图表的层次感和可读性。
2. 国家与编程语言分布（chart\_08）·功能：以旭日图展示不同国家的用户使用编程语言的分布情况，呈现国家与编程语言之间的层次关系和占比。·实现方式：统计每个国家用户使用的编程语言及其数量，使用Plotly的Sunburst图表类型绘制旭日图，根据用户数量设置颜色渐变，从红色到紫色，展示不同国家和编程语言组合的占比情况。通过自定义悬停信息样式，在鼠标悬停时显示详细信息。

（chart7：Top10组织与编程语言关系

chart8：国家与编程语言分布）

1. 用户多维度3D分布（chart\_09）·功能：以3D散点图展示用户在关注者数量、仓库数量、2024年贡献数和Stars数四个维度上的分布情况，帮助分析用户在多个方面的综合特征。·实现方式：将相关数据转换为数值型，使用Plotly的Scatter3d图表类型绘制3D散点图，每个点的位置表示用户在三个维度上的数值，点的大小根据Stars数进行对数缩放，颜色根据Stars数设置渐变，从橙色到绿色，同时添加颜色条方便用户理解颜色与Stars数的对应关系。通过设置相机视角，提供最佳的可视化效果。
2. 用户影响力分布（chart\_10）·功能：通过计算用户的综合影响力分数（考虑关注者数量、贡献数、仓库数量和注册年限等因素），以趋势图展示用户影响力随关注者排名的变化情况，同时以散点图展示每个用户的影响力分数和排名。·实现方式：定义函数计算影响力分数，对数据进行归一化处理，计算加权总分得到影响力分数。使用Plotly的Scatter图表类型绘制趋势线和散点图，趋势线展示整体影响力趋势，散点颜色根据影响力分数设置渐变，同时添加颜色条显示分数范围，悬停时显示用户详细信息。

（chart9：用户多维度3D分布

chart10：用户影响力分布）

1. 可视化界面设计与交互功能
2. 界面布局：使用Streamlit构建Web应用，通过创建多列布局（如两列或多列布局），合理安排各个可视化图表的展示位置，使页面布局简洁美观，便于用户查看和比较不同图表。
3. 交互功能·悬停提示：在图表中设置悬停提示功能，当用户鼠标悬停在数据点、柱状图、地图标记等元素上时，显示详细的信息，如用户名称、具体数值、地理位置等，帮助用户更好地理解数据。·缩放和平移：对于部分图表（如地图、折线图等），启用缩放和平移功能，用户可以通过鼠标滚轮或手势操作，放大或缩小图表，查看局部细节或整体趋势；平移图表可以查看不同区域的数据。·筛选和排序：提供筛选和排序功能，用户可以根据自己的需求选择特定的数据范围或对数据进行排序，例如按照关注者数量、贡献数等进行排序，以便更深入地分析数据。
4. 可视化结果分析
5. 地理分布特征

从世界地图可视化结果可以看出，Github关注量Top200用户主要分布在少数几个国家，如美国、中国、英国、德国等科技发达地区，这些地区的用户数量较多，在地图上呈现明显的热力集中区域。同时，通过用户位置标记可以发现，部分城市（如旧金山、纽约、北京、上海等）聚集了较多的高关注量用户，可能与当地的科技产业发展、教育资源丰富以及开源文化氛围浓厚等因素有关。

1. 关注者与贡献数关系

关注者数量和贡献数对比图表显示，部分用户关注者数量众多，但2024年贡献数相对较少，可能是一些知名开发者或项目负责人，其影响力主要体现在社交方面；而另一些用户虽然关注者数量较少，但贡献数较高，可能是活跃的开发者，专注于实际代码贡献。整体趋势上，关注者数量和贡献数之间存在一定的正相关关系，但并非完全线性相关，说明在Github社区中，用户的影响力来源具有多样性。

1. 用户国家分布情况

用户国家分布柱状图清晰展示了不同国家在Top200用户中的占比情况。美国用户占比较大，反映了其在全球开源领域的主导地位；中国紧随其后，显示中国开发者在Github上的活跃度和影响力不断提升。其他国家如英国、德国、印度等也有一定比例的用户，体现了全球范围内开源社区的多元性。

1. 编程语言偏好

最受欢迎语言Top10饼图揭示了当前Github用户群体中流行的编程语言。例如，某些语言在多个领域广泛应用，其在Top200用户中的使用频率较高；而一些新兴语言可能在特定领域或小众群体中受到青睐，也在图表中有所体现。通过分析编程语言偏好，有助于开发者了解行业技术趋势，选择合适的编程语言进行学习和项目开发。

1. 仓库数量分布

仓库数量分布直方图展示了用户在项目创建方面的活跃度差异。大部分用户的仓库数量集中在一定范围内，少数用户拥有较多的仓库，可能是经验丰富、项目众多的开发者或开源项目团队。该分布情况有助于了解Github平台上项目创建的整体规模和活跃度层次结构。

1. 加入Github年份趋势

加入Github年份分布图表呈现了用户加入平台的历史趋势。早期加入的用户数量相对较少，随着时间推移，用户数量逐渐增加，近年来增速有所变化，可能与开源文化的普及、行业发展趋势以及Github平台自身的推广和优化等因素相关。通过分析加入年份趋势，平台管理者可以评估不同阶段的用户增长情况，制定相应的发展策略。

1. 组织与编程语言关系

Top10组织与编程语言关系的桑基图展示了组织与编程语言之间的复杂关系。一些大型组织使用多种编程语言，体现了其在多技术领域的布局和项目需求多样性；同时，某些编程语言在多个组织中广泛应用，表明其通用性和重要性。该可视化结果有助于组织了解自身技术栈与行业趋势的匹配度，以及不同组织之间在技术选择上的共性和差异。

1. 国家与编程语言分布

国家与编程语言分布旭日图呈现了不同国家用户在编程语言使用上的偏好差异。一些国家在特定编程语言上具有较高的使用比例，可能与该国的教育体系、产业结构以及技术发展方向有关。例如，某些国家在新兴技术领域的编程语言应用较为领先，而另一些国家则在传统领域的编程语言使用上更为集中。通过分析该图表，能够深入了解全球范围内不同国家在开源技术领域的特点和优势。

1. 用户多维度综合特征

用户多维度3D分布散点图展示了用户在关注者数量、仓库数量、2024年贡献数和Stars数四个维度上的综合分布情况。可以观察到不同用户在各个维度上的表现差异，一些用户在多个维度上都表现出色，位于图表的特定区域；而另一些用户可能在某个维度上具有突出优势，但在其他维度上相对较弱。通过该可视化，能够更全面地评估用户的综合影响力和活跃度，为识别核心用户、发现潜在优秀开发者提供依据。

1. 用户影响力分布

用户影响力分布趋势图和散点图结合，清晰展示了用户影响力随关注者排名的变化情况。影响力分数较高的用户通常位于趋势线的上方，他们在关注者数量、贡献数、仓库数量和注册年限等方面综合表现优秀；而影响力分数较低的用户则分布在趋势线下方，可能需要在某些方面进一步提升。通过分析该图表，能够深入了解Github社区中用户影响力的分布结构，为平台制定用户激励机制、促进社区发展提供参考。

1. 关键AI对话内容

（注：下文“对话”中“A”表示Author，“C”表示Cursor Chat）

1. 初始设计

对话：

A：同一目录下有一个文件@person\_data\_top200.json，内容是Github关注量排名top200的个人用户信息，请据此在test.py中编写代码，实现Github关注度Top200个人用户信息数据大屏。数据大屏应包含以下信息（图表）：  
Chart1: 水平双向柱状图，followers & contributions（展示关注量排名和贡献情况，降序，可滑动）  
Chart2: 世界地图，loc（展示Top200用户地区分布，坐标和热点图）  
Chart3: 饼状图，languages（展示最热门的语言分布和比例）  
Chart4: 饼状图，is\_pro（展示Github Pro升级情况）  
Chart5: 柱状图，repo\_num（展示Top200用户仓库数量）  
Chart6: 柱状图，join\_date（展示Top200用户加入Github的年限情况）

C：以下是一个示例代码，展示如何在 test.py 中实现您所要求的 GitHub 关注度 Top 200 个人用户信息数据大屏。请确保您已经安装了所需的库，例如 pandas、matplotlib 和 plotly。

# test.py

import json

import pandas as pd

import matplotlib.pyplot as plt

import plotly.express as px

# 读取数据

with open('person\_data\_top200.json', 'r', encoding='utf-8') as f:

data = json.load(f)

# 将数据转换为 DataFrame

df = pd.DataFrame(data)

# Chart 1: 水平双向柱状图，followers & contributions

def chart1():

df['followers'] = df['followers'].str.replace('k', '').astype(float) \* 1000

df['contributions\_in\_2024'] = df['contributions\_in\_2024'].astype(int)

df\_sorted = df.sort\_values(by=['followers', 'contributions\_in\_2024'], ascending=False)

plt.figure(figsize=(10, 6))

plt.barh(df\_sorted['name'], df\_sorted['followers'], color='blue', label='Followers')

plt.barh(df\_sorted['name'], df\_sorted['contributions\_in\_2024'], color='orange', label='Contributions', alpha=0.5)

plt.xlabel('Count')

plt.title('Top 200 GitHub Users: Followers & Contributions')

plt.legend()

plt.show()

# Chart 2: 世界地图，loc

def chart2():

fig = px.choropleth(df, locations="loc", locationmode='country names',

title='Top 200 Users Location Distribution')

fig.show()

# Chart 3: 饼状图，languages

def chart3():

languages = df['languages'].explode().value\_counts()

plt.figure(figsize=(8, 8))

plt.pie(languages, labels=languages.index, autopct='%1.1f%%', startangle=140)

plt.title('Most Popular Languages Distribution')

plt.axis('equal')

plt.show()

# Chart 4: 饼状图，is\_pro

def chart4():

is\_pro\_counts = df['is\_pro'].value\_counts()

plt.figure(figsize=(8, 8))

plt.pie(is\_pro\_counts, labels=is\_pro\_counts.index, autopct='%1.1f%%', startangle=140)

plt.title('GitHub Pro Upgrade Status')

plt.axis('equal')

plt.show()

# Chart 5: 柱状图，repo\_num

def chart5():

df['repo\_num'] = df['repo\_num'].astype(int)

plt.figure(figsize=(10, 6))

plt.bar(df['name'], df['repo\_num'], color='green')

plt.xlabel('User')

plt.ylabel('Number of Repositories')

plt.title('Top 200 Users Repository Count')

plt.xticks(rotation=90)

plt.show()

# Chart 6: 柱状图，join\_date

def chart6():

df['join\_date'] = df['join\_date'].astype(int)

join\_counts = df['join\_date'].value\_counts().sort\_index()

plt.figure(figsize=(10, 6))

plt.bar(join\_counts.index, join\_counts.values, color='purple')

plt.xlabel('Join Year')

plt.ylabel('Number of Users')

plt.title('Top 200 Users Join Year Distribution')

plt.show()

# 调用所有图表函数

chart1()

chart2()

chart3()

chart4()

chart5()

chart6()

1. 结论与展望
2. 研究成果总结

本研究通过对Github关注量Top200用户数据的收集、清洗、转换和可视化分析，成功实现了多维度数据的直观展示。从地理分布、贡献情况、技术偏好等多个方面深入剖析了用户特征，为理解Github用户群体的行为模式和平台生态提供了丰富信息。通过多种可视化图表的设计与实现，展示了Python在数据可视化领域的强大功能和应用灵活性，为相关研究和实践提供了有益参考。

1. 研究局限性

·数据完整性：尽管努力获取了多方面的用户数据，但仍可能存在部分数据未被完全收集或因平台限制无法获取的情况，这可能影响分析结果的全面性。例如，一些用户的历史贡献数据可能不完整，导致对其长期贡献趋势的评估不够准确。·分析维度局限性：本研究主要关注了用户的部分关键维度信息，对于其他潜在重要因素（如用户所属行业、项目类型、协作模式等）未进行深入分析。这些因素可能对用户行为和影响力产生重要影响，未来研究可进一步拓展分析维度。·可视化方法优化空间：在可视化设计过程中，虽然尽力选择了合适的图表类型和交互功能，但仍存在一些可优化之处。例如，对于复杂的桑基图和旭日图，可能需要进一步改进布局和交互方式，以提高用户对数据关系的理解效率；部分图表在大数据量情况下的性能表现有待提升。

1. 未来研究方向

·拓展数据来源与维度：进一步整合更多Github平台数据以及外部相关数据，如项目代码质量评估数据、用户在其他开源平台的活动数据等，从更广泛的视角深入分析用户行为和平台生态。同时，挖掘更多用户行为相关维度，如代码审查活动、社区参与度等，构建更全面的用户画像。·深入分析用户行为模式：基于更丰富的数据，运用数据挖掘和机器学习技术，深入研究用户行为模式和影响力传播机制。例如，预测用户未来的贡献趋势、识别潜在的高影响力用户群体、分析不同用户行为对项目成功的影响等，为平台运营和开发者决策提供更精准的支持。·优化可视化技术与交互体验：持续探索新的可视化技术和方法，如结合虚拟现实（VR）/增强现实（AR）技术提供沉浸式数据可视化体验、利用动画效果展示数据动态变化过程、开发更智能的交互功能（如语音控制、自动推荐分析视角等），以提升用户对复杂数据的理解和探索能力。·跨平台与跨领域比较研究：将Github用户数据与其他类似开源平台或相关领域（如学术界的科研合作网络、企业内部的代码管理平台等）的数据进行对比分析，探索不同平台和领域之间用户行为、技术偏好和社区结构的异同点，为跨平台协作、知识共享和技术创新提供新的思路和方法。

1. 对计算机可视化领域的贡献

本研究通过实际项目展示了如何运用Python语言和相关可视化库，对复杂的Github用户数据进行全面、深入的可视化分析。在数据处理方面，提供了一套完整的数据清洗、转换和整合方法，为处理类似结构的数据提供了参考范例。在可视化设计上，通过多种定制化图表的实现，展示了如何根据数据特点和分析需求选择合适的可视化类型，并通过交互功能增强用户对数据的探索能力。研究成果有助于推动计算机可视化技术在开源社区分析、软件开发项目管理、开发者行为研究等领域的应用，为相关领域的研究人员和从业者提供了实践经验和技术借鉴，促进了计算机可视化领域与实际应用场景的深度融合。

综上所述，本研究在Github关注量Top200用户数据可视化分析方面取得了一定成果，但仍有诸多可拓展和优化的空间。未来研究将朝着更全面、深入、智能的方向发展，不断挖掘数据价值，提升可视化效果和用户体验，为开源社区和软件开发领域的发展提供有力支持。